ملاحظات هامة :

1. لا يشترط استيعاب كل نقطة في هذا النموذج في معيار النجاح هنا بل يعفى عن بعض الإغفالات والأخطاء ، فلو نسي مثلا بعض بنى أو تقنيات البرمجة ، أو بعض الأمثلة من النظام أو بعض مبادئ التصميم أو معايير الجودة فلا بأس ، ولكن يفضل أن ينبه المعلم الطالب إلى الأمور الأساسية .
2. يفضل عدم إدراج رموز المعايير في التقرير (مثل AP1,AP2 ...) ويمكن الاكتفاء بالعنوان .
3. لمن أراد ان يتقدم للتفوق والامتياز، لا يشترط فصل الفقرات إلى نجاح وتفوق وتميز بل يمكن دمجها معاً وهو الأفضل .
4. يوجد أيضا أمثلة أخرى في التطبيق على مهارات التفكير الحاسوبي أو مبادئ المنطق البرمجي أو غيرها ولكن ذكرت أمثلة فقط للتوضيح .
5. لن أطبق مهارات التفكير الحاسوبي على أكثر من نظام ، والمطلوب نظام واحد كما هو في المواصفات والذي يعتبر المرجع ، كما بُلغنا سابقاً .
6. يجب كتابة المراجع بطريقة صحيحة ، خاصة عند كتابة رابط موقع فيجب أن يكون فعال ويشير إلى المعلومات الصحيحة المتعلقة بالموضوع .
7. دائما في المقارنة يفضل عمل جداول كما في هذا النموذج.
8. يفضل عمل خرائط ورسومات لزيادة جودة التقرير .
9. يمكن فصل المزايا والعيوب بجداول أخرى ، لكن يفضل اتباع الطريقة التي في هذا النموذج لسرعة التقييم وتقليل عدد الصفحات .
10. في جدول نماذج البرمجة يكتفى بالشرح لإنه لم يطلب تحليل أو تقييم لنماذج البرمجة وإنما للغات البرمجة وغيرها .
11. في جدول المقارنة بين لغات البرمجة للتميز ، يفضل كتابة فقرة كاملة تشمل تحليل نقاط القوة والضعف بطريقة ضمنية ثم بعد ذلك يتم تقييمها (كما في النموذج) ، ويمكن الشرح ثم ذكر نقاط القوة ونقاط الضعف بطريقة صريحة .

Nejj Kamamiji T Ashraf Ghanem

طريقة إعداد تقرير لمهمة أ _ البرمجة

- ❖ **طبعا قبل كل شي يدرج الطالب الترويسة والتي تشمل اسمه ورقمه ويفضل بالإنجليزي مع شعار أو كلمة BTEC.**
- ❖ **يكتب عنوان المهمة بخط كبير وعريض .**
- ❖ **يعمل فهرس .**
- ❖ **يبدأ بمقدمة عامة (ممكن أن يشير فيها إلى معلومات عن البرنامج النموذجي والمشاكل والاحتياجات المحددة فيه والمراد حلها ، لكنه ليس ضروري).**

AP1 : شرح كيف يتم تطبيق مهارات التفكير الحسابي في إيجاد

(ملاحظة : لم أقم بشرح وتطبيق هذا المعيار على أكثر من نظام ، بل على نظام واحد وهو المكتبة)

- ❖ **يعرف التفكير الحاسوبي وأهميته في البرمجة .**
- ❖ **يشرح عن المهارات الأربعة بشكل عام** (ماذا تفعل هذه المهارة) مع التطبيق (على نظام المكتبة مثلا) :

أ- التفكيك : "هو تقسيم المشكلة إلى أجزاء أو مشكلات أصغر بحيث ..الخ ، كما في الكتاب " .

- **التطبيق على النظام :** من خلال فحص النظام نلاحظ أنه تم تقسيم المشكلة إلى مشكلات فرعية مثل:
  - إضافة كتاب جديد ، باستخدام دالة AddBook() :
  - استعارة الكتاب : LoanBook()
  - التنقل بين أجزاء النظام : الدالة Main()
  - ...الخ .

Najl Kamamji + Ashraf Ghanem

ب. تعرُّف الأنماط : " هو القدرة على تحديد السمات المشتركة ...الخ ، كما في الكتاب "

- **التطبيق على النظام:** نلاحظ وجود سمات وخطوات متكررة لكل عملية من العمليات التالية :
  - **نمط التحقق:** نلاحظ أن هناك نمطاً (if (!books.ContainsKey(id)) متكرراً يتكرر في دالتي: LoanBook() و ReturnBook()
  - **نمط التكرار :** Foreach .....
  - **نمط هيكلية (بنية) الكتاب:** **(**Class Book**)** ....

ج. تعميم النمط وتجريده : هو تصفية المعلومات غير الضرورية والتركيز فقط على المعلومات المطلوبة لحل المشكلة.

- **التطبيق على النظام:** نلاحظ بدلاً من التعامل مع كل خاصية على حدة، تم تجميعها في كلاس واحد وأيضا أخفى التفاصيل الغير ضرورية مثل تاريخ الإصدار ، دار النشر ،....الخ ، كالتالي :

```
class Book
{
    public string Title;
    public int TotalCopies;
    public int AvailableCopies;
}
```

مع ملاحظة انه يمكن تعميم هذا الكلاس لجميع الكتب أو تعميمه لفئات تشبهه مثل المجلات الأقراص ...الخ .

د- تمثيل عناصر المشكلة : هنا يتم تحديد الثوابت ،المتغيرات، المدخلات ، المخرجات، والعمليات وهياكل البيانات المناسبة لتمثيل الحل (كخوارزمية للحل) .

- **التطبيق على النظام:** نلاحظ وجود كل ما ذكر في البرنامج تقريبا كالتالي :
  - **الثوابت :** تم استخدام const string DATA_FILE لتمثيل اسم الملف، لأنه قيمة ثابتة لا تتغير أثناء تشغيل البرنامج.
  - **المتغيرات :** مثل ID و Total لتمثيل مدخلات المستخدم المتغيرة.
  - **هياكل البيانات :** مثل Stack لتمثيل آخر الكتب المسترجعة و ...الخ .
  - **العمليات** : مثل الدوال التي تمثل استعارة كتاب و....الخ .

Naji Kamami + Ashraf Ghanem

- ...الخ .

لمعيار التفوق AM1 :يجب القيام بالآتي :

❖ **تحليل مهارات تأثير التفكير الحاسوبي الأربعة وذلك بذكر تأثير كل مهارة على تصميم البرامج ، بعد شرح المهارة يكتب :**

- **مثلا على التفكيك** : "وهذا التفكيك يسمح بتقسيم البرنامج إلى دالات بحيث يساعد المبرمجين بالعمل بشكل متوازٍ مع كل قسم او دالة ثم بعد ذلك تجميعهم معا "،كما هو موجود في النظام.
- **التعرف على الأنماط** : التعرف على هذا النمط يمنعنا من كتابة كود مكرر في البرنامج ، وبدلاً من ذلك، نخطط لتصميم قالب واحد مشترك (مثل فئة أو دالة) يمكن إعادة استخدامها.
- **ثم يكمل باقي المهارات**

ولمعيار الامتياز AD1 :يجب القيام بالآتي :

❖ **يقيم مهارات تأثير التفكير الحاسوبي الأربعة وذلك بذكر تأثير كل مهارة على تصميم البرامج والتطبيقات المطورة (جودة المنتج النهائي) ، بعد الشرح والتحليل يكتب:**

- **مثلا على التفكيك** : " يؤثر على قابلية الصيانة حيث يعتبر العامل المهم في صيانة النظام، فإذا حدث خطأ في دالة حساب مخزون الكتب مثلا ، يمكن للمبرمج الذهاب مباشرة إلى الدالة أو الفئة المتعلقة بها لإصلاحها دون الحاجة للبحث في الكثير من أسطر . الكود المتداخل و بدون التفكيك، يصبح البرنامج كتلة واحدة مما يجعل أي تعديل بسيط محفوفاً بالمخاطر وقد يؤدي لتعطل النظام بالكامل.

  كما تؤثر على الموثوقية بحيث يسمح التفكيك باختبار كل دالة او فئة على حدة قبل دمجها مع باقي أجزاء البرنامج مما يرفع موثوقية المنتج النهائي بشكل كبير.

- **ثم يكمل باقي المهارات .**

❖ **وبعد الانتهاء يمكن كتابة حكم نهائي لجميع المهارات مثلا :**

أن تكامل مهارات التفكير الحاسوبي يمثل الركيزة الأساسية لضمان جودة البرمجيات؛ حيث يتضافر التفكيك وتصميم الخوارزميات (تمثيل عناصر المشكلة) لتعزيز الموثوقية وسرعة

Naji Kanamji + Ashraf Ghanem

الأداء، بينما يضمن تعرّف الأنماط والتجريد كفاءة الكود وقابليته للصيانة والتوسع المستقبلي. هذا الترابط يعمل توازناً دقيقاً بين متانة البنية البرمجية وسهولة استخدامها، مما يحقق أعلى معايير الجودة للنظام.

لجميع المعايير : شرح وتحليل وتقييم كيفية تطبيق مبادئ البرمجة

- **يفضل أن يشرح عن البرامج وما يتعلق بها من مفاهيم أساسية (مثل : البرمجة ، لغات البرمجة وانواعها (عالية أو منخفضة المستوى)، المترجم ، المفسر ...الخ .**
- **يشرح عن نماذج البرمجة المتاحة مع ما يمثلها من لغات برمجة ، أو العكس (أي مجموعة لغات البرمجة المتاحة وإلى أي نموذج تنتمي (لكن سنعتمد طريقة الكتاب وعمل جدول لذلك) كالتالي :**

النموذج	اللغات التي تمثلها	ميزاتها وخصائصها
الإجرائي	**Basic ,C,Fortran**	هي طريقة في البرمجة تتم من خلال كتابة تعليمات - تتركز في الغالب داخل دوال أو إجراءات - على شكل خطوات محددة ومتسلسلة لحل مشكلة معينة . حيث تمتاز بـ .....
الكائنية		
الموجهة بالأحداث		
الآلة		
الترميز		
البرمجة النصية		

- **يقارن مقارنة تقييمية (تشمل التحليل والشرح) بين اللغات المختلفة (ويختار 3 حسب طلب الوزارة) من حيث العوامل المذكورة في الجدول :**

Naji Kamanji - Ashraf Ghanem

المعيار	Python	C#	(VB.NET)
الأداء	أقل نسبيًا، لأنها لغة مفسّرة (interpreted) الأداء أقل من C# ، لكن يمكن تحسينه باستخدام مكتبات جاهزة مثل NumPy	أداء عالي نسبيا لإنها لغة مجمعة (Compiled) تعمل على بيئة .NET، مما يجعلها سريعة التنفيذ وقادرة على التعامل مع الخوارزميات المعقدة بكفاءة عالية ، لكن "جمع البيانات المهملة" قد تسبب في تقليل الأداء إذا كانت البيانات ضخمة جداً.	متوسطة الأداء، وقريبة من C# لإنها تعمل على .NET Framework وهي نفس البيئة التشغيلية الخاصة بـ C#، مما يعني أنها تقدم أداءً مقبولاً لمعظم تطبيقات الأعمال ، لكنها هي أبطأ من C# .
التقييم : (C#) : تعتبر الخيار الأمثل للتطبيقات التي تتطلب كفاءة عالية في استغلال المعالج وسرعة التنفيذ يأتي من كونها لغة مجمعة (Compiled) وصرامة تحديد نوع البيانات، مما يسمح للمترجم باكتشاف الأخطاء وتحسين الكود قبل التشغيل. (Python) : تعتبر الأقل كفاءة من حيث "وقت المعالج" واستهلاك الذاكرة (بسبب طبيعتها المفسرة والديناميكية) ولكن يمكن تعويض السرعة بالاعتماد على المكتبات الجاهزة والمعمولة بلغات سريعة مثل C . (VB.NET) : تعتبر الخيار المتوازن حيث تقدم أداءً تقنياً مطابقاً تقريباً لـ C# (لكونها تعتمد على نفس البيئة)، ولكن "أداؤها" في سوق العمل يتراجع بسبب انحسار شعبيتها وتوجه المطورين نحوC#.فهي خيار جيد فقط إذا كان الفريق يفضل قراءة الكود الشبيه باللغة الإنجليزية وتطوير تطبيقات مكتبية سريعة.			
الأجهزة الخاصة المستخدمة			
التقييم :			
مجالات التطبيق المفضلة			
التقييم :			
سهولة التطوير			
التقييم :			
وقت التطوير			
التقييم :			
الأجهزة			

Nati Kamjy Ashr

والبرامج اللازمة للتطوير والتشغيل			

التقييم العام :

تتميز C# بأداء عالٍ وكفاءة في استهلاك الموارد مما يجعلها مثالية للألعاب والأنظمة المؤسسية فإذا كان الهدف هو سرعة الأداء فإن C# هي الأفضل ، بينما توفر VB.NET سهولة قراءة الكود لتطبيقات ويندوز المكتبية رغم طول جملها. في المقابل، تتفوق Python في سرعة التطوير (Development Time) ومجالات الذكاء الاصطناعي لسهولتها وتعمل عبر مفسّر (Interpreter) على مختلف الأجهزة دون متطلبات عتاد معقدة، لكنها الأبطأ في التنفيذ مقارنة باللغات المترجمة...الخ .

❖ **يقارن مقارنة تقييمية بين اللغات المختلفة من حيث البنى والتقنيات والتنفيذ المستخدمة (مع ضرب الأمثلة من نظام المكتبة) كالتالي :**

وجه المقارنة مع التوضيح	Python	#C	(VB.NET)
الإعلان عن المتغيرات : تخبر هذه العملية البرنامج بنوع البيانات التي سيخزنها المتغير وتخصص له الذاكرة.	لا حاجة لإعلان صريح؛ المتغير يُنشأ بالتعيين **القوة:** مرن وسريع التطوير بسبب عدم حاجة المبرمج لتحديد النوع **الضعف:** ديناميكي، قد يسبب أخطاء وقت التشغيل وقل أمانًا في المشاريع الكبيرة.	تعتمد على نمط C (C-Style): تحدد النوع أولاً ثم الاسم، وتستخدم الفاصلة المنقوطة : مثل : public string Title **القوة :** الموثوقية العالية. لإنها لغات "ذات كتابة قوية وصارمة و تقلل الأخطاء. **الضعف:** تتطلب وقتاً أطول في الكتابة و صياغة معقدة لإنها معرفة مسبقة دقيقة بأنواع البيانات وحم الذاكرة.	تعتمد على الكلمات المفتاحية: تستخدم كلمات إنجليزية واضحة مثل Dim و As.مثال: Dim Title As String. كما في C#
التقييم : **C#** هي الخيار الأفضل للأنظمة الكبيرة والمعقدة التي تتطلب أداءً عالياً وموثوقية ومتانة لأن صرامتها			

Sharif Ghanem

في الإعلان والتعيين تحمي البرنامج من الانهيار المفاجئ.(يفضل صياغتها بطريقة تدل على ربطها في نظام المكتبة) **VB.NET** : ممتازة لسرعة التطوير في بيئة ويندوز، خاصة للمبرمجين الذين يفضلون القراءة السهلة للكود، لكنها أقل شيوعاً الآن مقارنة بـ C#. **Python:** : تتفوق في سهولة التطوير والتعلم (Ease of Development). هي الأنسب للمشاريع السريعة، الذكاء الاصطناعي، والسكربتات، لكنها تتطلب اختباراً دقيقاً لتعويض غياب التدقيق الصارم للأنواع			
تعيين القيم : التعيين هو إعطاء قيمة لمتغير سواء كانت رقمية أو نصية أو غيرها			
التقييم :			
الإدخال			
الإخراج			
التقييم :			
العوامل المنطقية (إن وجدت)			
النصوص والسلاسل			
الاختيار (...,if,switch)			
التكرار (...for,while,foreach)			
الدالات / الإجراءات			
هياكل (تراكيب) البيانات (list,struct,stack,array,dictiona..,ry)			
معالجة الملفات			

hji + Ashraf Gh

			معالجة الأحداث (إن وجدت)
			التوثيقات
			الفئات (الكلاس) :

تقييم عام (للبنى والتقنيات) كما استخدمت في نظام المكتبة : تكتب فقرة تقييمية عامة مرتبطة بنظام المكتبة

تقييم كيفية استخدام مبادئ تصميم البرامج لإنتاج تطبيقات برمجية

- **يشرح ويحلل ويقيم مبادئ المنطق المطبقة على تصميم البرنامج (نظام المكتبة) :**

- **المنطق :** هو مجموعة المبادئ الموجودة في العناصر المختلفة في البرنامج والتي تسمح للبرنامج بالعمل حيث يحدد كيفية تصرف البرامج والإجراءات التي تقوم بها. وتشمل الآتي:

1- التكرار – هو منطق برمجي يتحكم في تكرار الكود ويحدد متى يجب أن ينتهي التكرار في أثناء تشغيل البرنامج .

- **التطبيق على النظام** : نلاحظ أنه تم تطبيق هذا المبدأ في أكثر من مكان في الكود مثل :

```
foreach (var b in books)
        {
            sw.WriteLine($"{b.Key}|{b.Value.Title}|{b.Value.TotalCopies}|
{b.Value.AvailableCopies}");
        }
```

المنطق هنا : طالما يوجد عنصر آخر للكتب، كرر نفس العملية ،أي عملية (طباعة تفاصيل الكتاب) على كل عنصر يتعلق بـ Books .

التحليل : استخدام التكرار جعل الكود مختصراً جداً. بدلاً من كتابة أمر طباعة لكل كتاب يدويًا أي سمح التكرار بمعالجة جميع البيانات بكود واحد.

التقييم : وهذا يحقق **الكفاءة والاستمرارية** للنظام حيث كود مختصر وشامل .

Aya Kamaniji F. Ashraf Ghanem

2- المنطق الرياضي - ..... تعريف مختصر :

- **التطبيق في النظام :** نلاحظ هذا بوضوح في المقارنات في الجمل الشرطية (if / switch) وجمل التكرار (while) وغيرها ،مثل :

```
if (!books.ContainsKey(id))
{
   Console.WriteLine("Book not found.");
   return;
}
```

الاستدلال المنطقي هنا : إذا لم يكن الكتاب موجودًا إذا لا يمكن إعارتُه أو إرجاعه.

التحليل :هذا بدوره يمنع البرنامج من الفشل والانهيار إذا أدخل المستخدم رقم تعريف خاطئ .

التقييم : وبالتالي : يحقق **الاستمرارية** في عمل النظام .

3- المنطق الديناميكي الافتراضي :.....الخ .

4- استخدام المجموعات: .....الخ

التقييم العام :

وظف الكود **التكرار** (while, foreach) بفاعلية لاستمرار تشغيل القائمة ومعالجة كافة العناصر ، بينما ضمن **المنطق الرياضي** (if, <=) سلامة العمليات بمنع الإعارة عند نفاد النسخ. كما حقق **المنطق الديناميكي** انتقالاً صحيحاً لحالة الكتاب من "متاح" إلى "مُعار" والعودة ، مستنداً إلى مبدأ **المجموعات** عبر Dictionary لضمان تفرد معرفات الكتب وسرعة الوصول إليه ..الخ .

- **يشرح ويحلل ويقيم مبادئ التصميم والبرمجة التي تجعل البرنامج "عالي الجودة" ويلبي احتياجات المستخدم:**

يؤثر تصميم وتنفيذ الحل البرمجي على جودته من نواحٍ مختلفة مثل:

- **الكفاءة والأداء -** وهو قياس موارد النظام التي يستهلكها البرنامج مثل الذاكرة ووقت المعالجة ...الخ .

Naji Kamamji + Ashraf Ghanem

التطبيق من النظام : استخدام هياكل مناسبة لتسريع الأداء مثل stacks وQueue و Dictionary...

التحليل : عملية الوصول سريعة جداً. ومع ذلك، يتم تحميل كامل الملف للنظام في الذاكرة Load() ، وبالمختصر **التقييم :** أداء سريع في البحث والإعارة بفضل البحث باستخدام خوارزمية الهاش.

- **القابلية للصيانة -** هي سهولة تغيير الكود من قبل المبرمجين ...الخ .

التطبيق من النظام : الكود منظم باستخدام دوال وفئات(كلاسات) وكذلك التوثيقات بمختلف أنواعها ...

- **القابلية للنقل -** .....
- **الموثوقية -** .....
- **المتانة -** .....
- **سهولة الاستخدام -** .....

التقييم العام :
الكود في نظام المكتبة يحقق مستوى جيد من الجودة من حيث **الوضوح والكفاءة** باستخدام هياكل بيانات مناسبة ويوفر **موثوقية** مقبولة عبر التحقق من توفر النسخ قبل الإعارة ، لكن يعاني أحيانا النظام من ضعف في **المتانة**؛ حيث يفتقر لمعالجة الأخطاء عند إدخال بيانات غير رقمية. **المرونة** متوسطة بسبب دمج واجهة المستخدم مع منطق النظام، مما يقلل من **سهولة التوسع والقابلية للصيانة** تحتاج تعزيز عبر إضافة تعليقات ، و وحدات و إدارة البيانات جيدة لكنها محدودة بالملفات النصية فقط.
بشكل عام، الكود يلبي الاحتياجات الأساسية لنظام مكتبة صغير لكنه يحتاج تحسينات ليكون أكثر احترافية وقابلية للتوسع.

Naji Kenjami + Ashraf Ghanem

❖ **يكتب المراجع في نهاية التقرير :**